

EXPRESS MAIL NO. EL 746 760 130 US

DATE OF DEPOSIT

8/8/01



Our File No. 9281-4144  
Client Reference No. J US98135

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

In re Application of: )  
Takayuki Sugawara )  
Serial No. To Be Assigned )  
Filing Date: Herewith )  
For: Arithmetic Operation Unit Suitable )  
for Correcting Lost Data by )  
General-Purpose Computer )

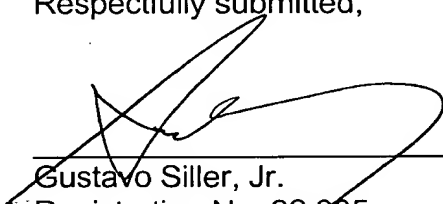
**SUBMISSION OF CERTIFIED COPY OF PRIORITY DOCUMENT**

Commissioner for Patents  
Washington, D.C. 20231

Dear Sir:

Transmitted herewith is a certified copy of priority document Japanese Patent Application No. 2000-243774, filed August 11, 2000 for the above-named U.S. application.

Respectfully submitted,

  
\_\_\_\_\_  
Gustavo Siller, Jr.  
Registration No. 32,305  
Attorney for Applicant

BRINKS HOFER GILSON & LIONE  
P.O. BOX 10395  
CHICAGO, ILLINOIS 60610  
(312) 321-4200

日 本 国 特 許 庁  
JAPAN PATENT OFFICE



別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出 願 年 月 日  
Date of Application:

2000年 8月11日

出 願 番 号  
Application Number:

特願2000-243774

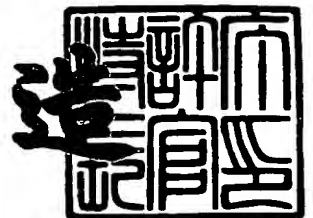
出 願 人  
Applicant(s):

アルプス電気株式会社

2001年 6月 4日

特 許 庁 長 官  
Commissioner,  
Japan Patent Office

及 川 耕 造



出証番号 出証特2001-3051919

【書類名】 特許願

【整理番号】 981212AL

【提出日】 平成12年 8月11日

【あて先】 特許庁長官殿

【国際特許分類】 G11B 20/18

【発明の名称】 演算処理装置

【請求項の数】 5

【発明者】

【住所又は居所】 東京都大田区雪谷大塚町 1 番 7 号 アルプス電気株式会社  
社内

【氏名】 菅原 孝幸

【特許出願人】

【識別番号】 000010098

【氏名又は名称】 アルプス電気株式会社

【代表者】 片岡 政隆

【代理人】

【識別番号】 100085453

【弁理士】

【氏名又は名称】 野▲崎▼ 照夫

【手数料の表示】

【予納台帳番号】 041070

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【ブルーフの要否】 要

【書類名】 明細書

【発明の名称】 演算処理装置

【特許請求の範囲】

【請求項 1】 数 1 で表されるガロア体上の原始多項式  $G(x)$  の  $G(x) = 0$  の元を  $\alpha$  としたときに、 $\alpha^i$  を乗算する演算処理装置であって、

【数 1】

$$G(x) = g_m x^m + g_{m-1} x^{m-1} + g_{m-2} x^{m-2} + \dots + g_{p+1} x^{p+1} + g_p x^p + \dots + g_0$$

乗算前の元を  $i$  ビットシフトするシフト演算部と、 $\alpha$  の乗数を  $i$  としたときに  $2^i$  個の要素のルックアップテーブルを参照する参照部とを有することを特徴とする演算処理装置。

【請求項 2】 前記数 1 で表されるガロア体上の原始多項式  $G(x)$  の  $G(x) = 0$  の元を  $\alpha$  としたときに、以下の数 2 で示されるガロア体上の元  $U$  を基に  $U \cdot \alpha^i$  を演算する演算処理装置であって、

【数 2】

$$U = \alpha^n u_n + \alpha^{n-1} u_{n-1} + \dots + \alpha^2 u_2 + \alpha^1 u_1 + u_0$$

前記  $U$  の元を  $i$  ビットシフトするシフト演算部と、 $U$  の最下位の  $i$  ビットに応じて、 $2^i$  個の要素のルックアップテーブルを参照する参照部との排他的論理和をとることを特徴する演算処理装置。

【請求項 3】 データを  $D_1, D_2, \dots, D_k$  としたときに以下の数 3 で示されるエラー検査シンボル  $E_0, E_1, E_2, \dots, E_{n-k-1}$  が演算される請求項 2 記載の演算処理装置。

【数 3】

$$\begin{aligned} D_1 + D_2 + D_3 + \cdots + D_{k-1} + D_k &= E_0 \\ \alpha^k D_1 + \alpha^{k-1} D_2 + \alpha^{k-2} D_3 + \cdots + \alpha^2 D_{k-1} + \alpha D_k &= E_1 \\ \alpha^{\omega^2} D_1 + \alpha^{(\omega^{-1})^2} D_2 + \alpha^{(\omega^{-2})^2} D_3 + \cdots + \alpha^4 D_{k-1} + \alpha^2 D_k &= E_2 \\ &\vdots \\ \alpha^{\omega^{n-k-1}} D_1 + \alpha^{(\omega^{-1})^{n-k-1}} D_2 + \cdots + \alpha^{n-k} D_{k-1} + \alpha^{n-k-1} D_k &= E_{n-k-1} \end{aligned}$$

【請求項 4】 データをデコードしたときに、以下の数 4 の演算を行ってシンボル  $S_0, S_1, S_2, \dots, S_{n-k-1}$  を得る請求項 3 記載の演算処理装置。

【数 4】

$$\begin{aligned} D_1 + D_2 + D_3 + \cdots + D_{k-1} + D_k + E_0 &= S_0 \\ \alpha^k D_1 + \alpha^{k-1} D_2 + \alpha^{k-2} D_3 + \cdots + \alpha^2 D_{k-1} + \alpha D_k + E_1 &= S_1 \\ \alpha^{\omega^2} D_1 + \alpha^{(\omega^{-1})^2} D_2 + \alpha^{(\omega^{-2})^2} D_3 + \cdots + \alpha^4 D_{k-1} + \alpha^2 D_k + E_2 &= S_2 \\ &\vdots \\ \alpha^{\omega^{n-k-1}} D_1 + \alpha^{(\omega^{-1})^{n-k-1}} D_2 + \cdots + \alpha^{n-k} D_{k-1} + \alpha^{n-k-1} D_k + E_{n-k-1} &= S_{n-k-1} \end{aligned}$$

【請求項 5】 前記シンボル  $S_0, S_1, S_2, \dots, S_{n-k-1}$  を用いてエラーの大きさを求める際、次の逆元参照テーブルを設けて、これを参照して前記エラーの大きさを求める請求項 4 記載の演算処理装置。

- a)  $\alpha^1, \alpha^2, \dots, \alpha^k,$
- b)  $1 + \alpha^1, 1 + \alpha^2, \dots, 1 + \alpha^k$

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、ガロア体上の元を高速に乗算できる演算処理装置に係り、特にデータの消失訂正を汎用の計算機を行うような場合に適した演算処理装置に関する。

【0002】

【従来の技術】

記録媒体へのデータの記録および再生の際などに行われるエラー訂正処理では、検査シンボル  $E_0, E_1, E_2, \dots$  が付加コードとして記録媒体にエンコードさ

れ、再生時にはデータ（ユーザシンボル）に前記検査シンボル  $E_0$ ,  $E_1$ ,  $E_2$ , …の排他的論理和をとったシンボル  $S_0$ ,  $S_1$ ,  $S_2$ , …が求められる。そしてこのシンボルを演算することにより、エラーの大きさが算出される。

## 【0003】

前記シンボル  $E_0$  と  $S_0$  は、単純な排他的論理和により計算が可能であるが、 $E_1$ ,  $S_1$  以降では、 $\alpha$  のべき乗算の計算を行う必要がある。ここで、ガロア体上の原始多項式  $G(x)$  が 0 となるときの元が  $\alpha$  である。

## 【0004】

記録再生装置などでは、ガロア体の専用演算部を持たないために、ルックアップテーブルを用いて前記検査シンボル  $E$  やデコード時のシンボル  $S$  を求めている。

## 【0005】

## 【発明が解決しようとする課題】

従来は、前記ルックアップテーブルとして 8 ビット (256)  $\times$  8 ビット (256) = 64 k バイトのテーブルを用いている。よって 8 ビットずつのデータに対しては、検査シンボル  $E$  やデコード時のシンボル  $S$  を求めることが可能である。しかし、現在では計算機のアクセス幅が 16 ビットや 32 ビット単位となっているため、16 ビット単位や 32 ビット単位でシンボルの演算を行うことが好ましい。

## 【0006】

しかし、16 ビット単位の演算では、16 ビット (64 k バイト)  $\times$  16 ビット (64 k バイト) = 4 G バイトとなり、前記のようなルックアップテーブルを構築することは実質的に不可能である。

## 【0007】

またべき乗表現の変換テーブルを使用することも考えられるが、この場合も 16 ビット単位で数百 k バイトオーダのテーブルが必要になる。

## 【0008】

本発明は上記従来の課題を解決するものであり、ユーザシンボルが何ビットであっても、ガロア体の元  $\alpha$  を用いたべき乗計算を高速に行え、例えば消失訂正を

高速に実現できるようにした演算処理装置を提供することを目的としている。

【0009】

【課題を解決するための手段】

本発明は、前記数1で表されるガロア体上の原始多項式 $G(x)$ の $G(x) = 0$ の元を $\alpha$ としたときに、 $\alpha^i$ を乗算する演算処理装置であって、

乗算前の元を $i$ ビットシフトするシフト演算部と、 $\alpha$ の乗数を $i$ としたときに $2^i$ 個の要素のルックアップテーブルを参照する参照部とを有することを特徴とするものである。

【0010】

また本発明は、前記数1で表されるガロア体上の原始多項式 $G(x)$ の $G(x) = 0$ の元を $\alpha$ としたときに、前記数2で示されるガロア体上の元 $U$ を基に $U \cdot \alpha^i$ を演算する演算処理装置であって、

前記 $U$ の元を $i$ ビットシフトするシフト演算部と、 $U$ の最下位の $i$ ビットに応じて、前記ルックアップテーブル参照結果との排他的論理和をとることを特徴するものである。

【0011】

また、データ（ユーザシンボル）を $D_1, D_2, \dots, D_k$ としたときに前記数3で示されるエラー検査シンボル $E_0, E_1, E_2, \dots, E_{n-k-1}$ が演算される。

【0012】

および／または、データ（ユーザシンボル）をデコードしたときに、以下の数4の演算を行ってシンボル $S_0, S_1, S_2, \dots, S_{n-k-1}$ を得ることができる。

【0013】

また、前記シンボル $S_0, S_1, S_2, \dots, S_{n-k-1}$ を用いてエラーの大きさを求める際、次の逆元参照テーブルを設けて、これを参照して前記エラーの大きさを求めることが可能である。

【0014】

- a)  $\alpha^1, \alpha^2, \dots, \alpha^k,$
- b)  $1 + \alpha^1, 1 + \alpha^2, \dots, 1 + \alpha^k$

【0015】

【発明の実施の形態】

本発明は、計算機のアクセス幅（例えば16ビットや32ビットなど）に一致する次元のガロア体上の原始多項式  $G(x) = 0$  の元を  $\alpha$  としたときに、高速な  $\alpha^i$  の乗算を行えるようにするものである。本発明の演算処理装置では、計算機のアクセス幅が大きくても高速な乗算が可能であり、従来のように8ビット単位でしかエラー訂正ができなかったという問題を解消することができる。

【0016】

以下の表1は、磁気ディスクなどに記録されるデータの1ブロック単位のフォーマットを示している。

【0017】



【表 1】

【表 1】

Packet No	Double Word 0	Double Word 1	Double Word 2		Double Word 1 2 7
0	Byte 0-3 (D <sub>1</sub> )	Byte 4-7	Byte 8-11	.....	Byte 508-511
1	Byte 0-3 (D <sub>2</sub> )	Byte 4-7	Byte 8-11	.....	Byte 508-511
.....	.....	.....	.....	.....	.....
63	Byte 0-3 (D <sub>63</sub> )	Byte 4-7	Byte 8-11	.....	Byte 508-511
64	ECC[0]	ECC[0]	ECC[0]	.....	ECC[0]
65	ECC[1]	ECC[1]	ECC[1]	.....	ECC[1]

【 0 0 1 8 】

上記表1の縦方向はPacket Noであり、0～63までが、ユーザブロックであり、ここにはデータ（ユーザシンボル）が含まれる。Packet Noの64と65がエラー訂正のための検査シンボル $E_0$ と $E_1$ が含まれる。この検査シンボル $E_0$ と $E_1$ がエラー訂正のためのコードである。

【0019】

Packet Noの0から63までのデータは32ビット（4バイト）のDouble Word単位で、Packet Noの0から63にかけて演算が施され、各Double Word単位で検査シンボル $E_0$ が演算されて、Packet No64内に記録される。また各Double Word単位で検査シンボル $E_1$ が演算されて、Packet No65内に記録される。

【0020】

記録媒体に表1で示すフォーマットでデータを記録する際に、各Double Word単位で検査シンボル $E_0$ が演算される。この演算の一般式を以下の数5に示す。以下の数5において、 $D_1, D_2, D_3, \dots, D_k$ は、各Double Word単位でのPacket Noの0から63までの例えばそれぞれ32ビットのユーザシンボルである。なお表1の場合、前記シンボル数 $k$ は63である。

【0021】

$E_0, E_1, E_2, \dots, E_{n-k-1}$ は検査シンボルであり、検査シンボル数は $n-k$ である。なお、表1は検査シンボル数が2の場合を示している。

【0022】

【数5】

$$\begin{aligned} D_1 + D_2 + D_3 + \dots + D_{k-1} + D_k &= E_0 \\ \alpha^k D_1 + \alpha^{k-1} D_2 + \alpha^{k-2} D_3 + \dots + \alpha^2 D_{k-1} + \alpha D_k &= E_1 \\ \alpha^{\omega^2} D_1 + \alpha^{(\alpha-1)^2} D_2 + \alpha^{(\alpha-2)^2} D_3 + \dots + \alpha^4 D_{k-1} + \alpha^2 D_k &= E_2 \\ &\vdots \\ \alpha^{(\alpha)^{n-k-1}} D_1 + \alpha^{(\alpha-1)^{n-k-1}} D_2 + \dots + \alpha^{n-k} D_{k-1} + \alpha^{n-k-1} D_k &= E_{n-k-1} \end{aligned}$$

【0023】

次に、記録媒体からデータをデコードするときには、以下の数6の演算が行われる。この演算によるシンボル $S_0, S_1, S_2, \dots, S_{n-k-1}$ により後に示すように、データのエラーの大きさが求められる。

【 0 0 2 4 】

【数 6】

$$\begin{aligned} D_1 + D_2 + D_3 + \cdots + D_{k-1} + D_k + E_0 &= S_0 \\ \alpha^k D_1 + \alpha^{k-1} D_2 + \alpha^{k-2} D_3 + \cdots + \alpha^2 D_{k-1} + \alpha D_k + E_1 &= S_1 \\ \alpha^{\omega^2} D_1 + \alpha^{(\alpha-1)^2} D_2 + \alpha^{\alpha-2^2} D_3 + \cdots + \alpha^4 D_{k-1} + \alpha^2 D_k + E_2 &= S_2 \\ &\vdots \\ \alpha^{\alpha^{n-k-1}} D_1 + \alpha^{(\alpha-1)^{n-k-1}} D_2 + \cdots + \alpha^{n-k} D_{k-1} + \alpha^{n-k-1} D_k + E_{n-k-1} &= S_{n-k-1} \end{aligned}$$

【 0 0 2 5 】

エンコードとデコード時での前記数 5 と数 6 は、シンボル数マイナス 1 までの  $\alpha^i$  乗算器があれば演算可能である。なお  $i$  は 0, 1, 2, 3, ...,  $k$  である。  
 なお、前記数 1 以下の各式における「+」は排他的論理和 (E x O R) である。

【 0 0 2 6 】

まず、前記数 5 と数 6 における検査シンボル  $E_0$  と  $S_0$  は排他的論理和により単純に計算可能であるが、 $E_1$  と  $S_1$  以降の計算は、べき乗算が必要である。図 1 はこの計算を行う演算回路の一例を示す。図 1 に示すように、レジスタ内に  $D_1$ ,  $D_2$ ,  $D_3$ , ...,  $D_k$  を格納しておき、それぞれに  $\alpha$ ,  $\alpha^2$ , ...,  $\alpha^{n-k-1}$  をべき乗算していくことにより、 $E_1$ ,  $E_2$ , ...,  $E_{n-k-1}$  および  $S_1$ ,  $S_2$ , ...,  $S_{n-k-1}$  の演算が可能である。

【 0 0 2 7 】

そこでこのべき乗算を高速化する本発明の計算方法および演算処理装置について説明する。

【 0 0 2 8 】

ガロア体  $GF(2)$  の原始多項式を以下の数 7 とする。

【 0 0 2 9 】

【数 7】

$$G(x) = g_m x^m + g_{m-1} x^{m-2} + g_{m-2} x^{m-2} + \cdots + g_{p+1} x^{p+1} + g_p x^p + \cdots + g_0$$

【 0 0 3 0 】

この演算処理装置の原理を簡単に説明できるように、ここではガロア体の原始

多項式を以下の数 8 を例として説明する。

【0031】

【数 8】

$$G_{\omega} = X^8 + X^4 + X^3 + X^2 + 1$$

【0032】

また、数 4 と数 5 において、ユーザシンボル  $D_1, D_2, D_3, \dots, D_k$  に対応するシンボルを 8 ビットの  $u_7, u_6, u_5, u_4, u_3, u_2, u_1, u_0$  とすると、 $U$  ( $u_7, u_6, u_5, u_4, u_3, u_2, u_1, u_0$ ) は、以下の数 9 となる。

【0033】

【数 9】

$$U = \alpha^7 u_7 + \alpha^6 u_6 + \alpha^5 u_5 + \alpha^4 u_4 + \alpha^3 u_3 + \alpha^2 u_2 + \alpha^1 u_1 + u_0$$

【0034】

また  $\alpha \cdot U$  は以下の数 10 のようになる。

【0035】

【数 10】

$$\alpha \cdot U = \alpha^8 u_7 + \alpha^7 u_6 + \alpha^6 u_5 + \alpha^5 u_4 + \alpha^4 u_3 + \alpha^3 u_2 + \alpha^2 u_1 + \alpha u_0$$

【0036】

上記数 10 に示される  $\alpha \cdot U$  は、数 9 の  $U$  の元において  $u_7, u_6, u_5, u_4, u_3, u_2, u_1, u_0$  を 1 ビットシフトさせて、以下の数 11 の通りとし、これに数 8 の原理多項式から得られた数 12 を加算（排他的論理和）したものに等しい。

【0037】

【数 1 1】

$$\alpha^7 u_6 + \alpha^6 u_7 + \alpha^5 u_4 + \alpha^4 u_3 + \alpha^2 u_1 + \alpha^1 u_0$$

【0 0 3 8】

【数 1 2】

$$\alpha^8 ( = \alpha^4 + \alpha^3 + \alpha^2 + 1 ) \cdot u_7$$

【0 0 3 9】

よってUから $\alpha \cdot U$ への演算では、Uの最下位ビットu 7が「1」の場合と「0」の場合において、2通りのルックアップテーブル（参照部，表 2）を、シフト演算部と共に設けることで、前記演算を高速に行うことができる。

【0 0 4 0】

【表 2】

【表2】

u7	ルックアップテーブルの値
0	0
1	$\alpha^4 + \alpha^3 + \alpha^2 + \alpha^1$

【0 0 4 1】

次に、 $\alpha^2 \cdot U$ は以下の数 1 3 の通りである。

【0 0 4 2】

【数13】

$$\alpha^2 \cdot U = \alpha^8 u_7 + \alpha^8 u_6 + \alpha^7 u_5 + \alpha^6 u_4 + \alpha^5 u_3 + \alpha^4 u_2 + \alpha^3 u_1 + \alpha^2 u_0$$

【0043】

これは、前記数9のUの元を2ビットシフトさせ、以下の数14を足した（排他的論理和）ものに等しい。

【0044】

すなわち、Uの元の下位2ビット値に応じた $2^2=4$ 通りのルックアップテーブル（表3）を用意しておき、Uの2ビットシフト演算との排他的論理和をとれば良い。

【0045】

【表3】

【表3】

u6, u7	ルックアップテーブルの値
0 0	0
0 1	$\alpha^4 + \alpha^3 + \alpha^2 + \alpha^1$
1 0	$\alpha (\alpha^4 + \alpha^3 + \alpha^2 + \alpha^1)$
1 1	$\alpha (\alpha^4 + \alpha^3 + \alpha^2 + \alpha^1) + \alpha^4 + \alpha^3 + \alpha^2 + \alpha^1$

【0046】

【数14】

$$\alpha^8 (= \alpha \cdot (\alpha^4 + \alpha^3 + \alpha^2 + 1)) \cdot u_7 + \alpha^8 (= \alpha^4 + \alpha^3 + \alpha^2 + 1) \cdot u_6$$

【0047】

シフト演算は、C P O に一般的に備わっており、またルックアップテーブルを小容量のメモリで実現できるから、汎用かつ高速な演算方法である。

【 0 0 4 8 】

以上は  $\alpha^3 \cdot U$ ,  $\alpha^4 \cdot U$  においても同じである。すなわち  $\alpha^i$  のためには、

- 1)  $i$  ビットのシフト演算部
- 2)  $2^i$  の数のルックアップテーブル (参照部)

を設けることにより高速演算が可能である。

【 0 0 4 9 】

次に、前記シンボル  $S_0, S_1, S_2, \dots, S_{n-k-1}$  によるエラー訂正について説明する。

【 0 0 5 0 】

前記記録媒体から再生したデータにエラーが無い場合には、数 6 においてシンボル  $S_0, S_1, S_2, \dots, S_{n-k-1}$  は全て 0 になる。またユーザシンボル中にエラーが生じた場合、エラーが生じている箇所が予め分かっている消失訂正においては以下のような演算でエラーの大きさを算出できる。

【 0 0 5 1 】

例えばユーザシンボル中のエラーが、後ろから  $i$  のポジションと  $j$  のポジションで生じているとすると、エラーの大きさを  $e_i, e_j$  とすると、これとシンボル  $S_0, S_1$  との関係は以下の数 1 5 のようになる。

【 0 0 5 2 】

【 数 1 5 】

$$e_i + e_j = S_0$$

$$\alpha^i e_i + \alpha^j e_j = S_1$$

【 0 0 5 3 】

上記数 1 5 から  $e_i, e_j$  を求めると以下の数 1 6 のようになるが、これは 2 元連立方程式として解くことができる。

【 0 0 5 4 】

【数16】

$$e_i = \frac{\alpha^j \cdot S_0 \cdot S_1}{\alpha^i + \alpha^j}$$

$$e_j = \frac{(\alpha^j \cdot S_0 \cdot S_1) \cdot \alpha^{-i}}{1 + \alpha^{j-i}}$$

【0055】

次に、検査シンボル $E_0$ に誤り $e_j$ があったとすると、以下の数17のようになり、エラーの大きさ $e_i$ は1次方程式で解くことができる。

【0056】

【数17】

$$e_i + e_j = S_0$$

$$\alpha^i e_i = S_1$$

【0057】

同様にユーザシンボルに3箇所のエラー $e_i$ ,  $e_j$ ,  $e_k$ があったときには、数18に示す3元連立方程式を解くことによりエラーの大きさ $e_i$  (数19)を求めることができる。

【0058】

【数18】

$$e_i + e_j + e_k = S_0$$

$$\alpha^i e_i + \alpha^j e_j = S_1$$

$$\alpha^{2i} e_i + \alpha^{2j} e_j = S_2$$

【0059】



【数 19】

$$e_i = \frac{\alpha^j \cdot \alpha^k \cdot S_0 \cdot \alpha^j \cdot S_1 \cdot \alpha^k \cdot S_1 + S_2}{(-\alpha^i + \alpha^j)(-\alpha^i + \alpha^k)}$$

$$e_j = \frac{(\alpha^j \alpha^k S_0 \alpha^j S_1 \alpha^k S_1 + S_2) \alpha^{-j-k}}{(1 - \alpha^{i-j})(1 - \alpha^{i-k})}$$

【0060】

同様にして  $e_j$ ,  $e_k$  を演算できる。

上記の演算において、原始多項式の次数が高くなるにしたがって逆元の計算が膨大なものになる。よって予め決まった値以内になるように変形しておいてテーブルルックアップ方式で逆元を参照することが速度的に有利である。この逆元に関しては、次の2種類のテーブル（参照部）を持てば十分である。

【0061】

a)  $\alpha^1, \alpha^2, \dots, \alpha^k$  ( $k$  はユーザーシンボル数)

b)  $1 + \alpha^1, 1 + \alpha^2, \dots, 1 + \alpha^k$  ( $k$  はユーザーシンボル数)

【0062】

【発明の効果】

以上のように本発明では、ガロア体の原始多項式を用いたエラー訂正用の演算を高速で行えるように成る。

【図面の簡単な説明】

【図 1】

$\alpha$  乗算回路の一例を示すブロック図

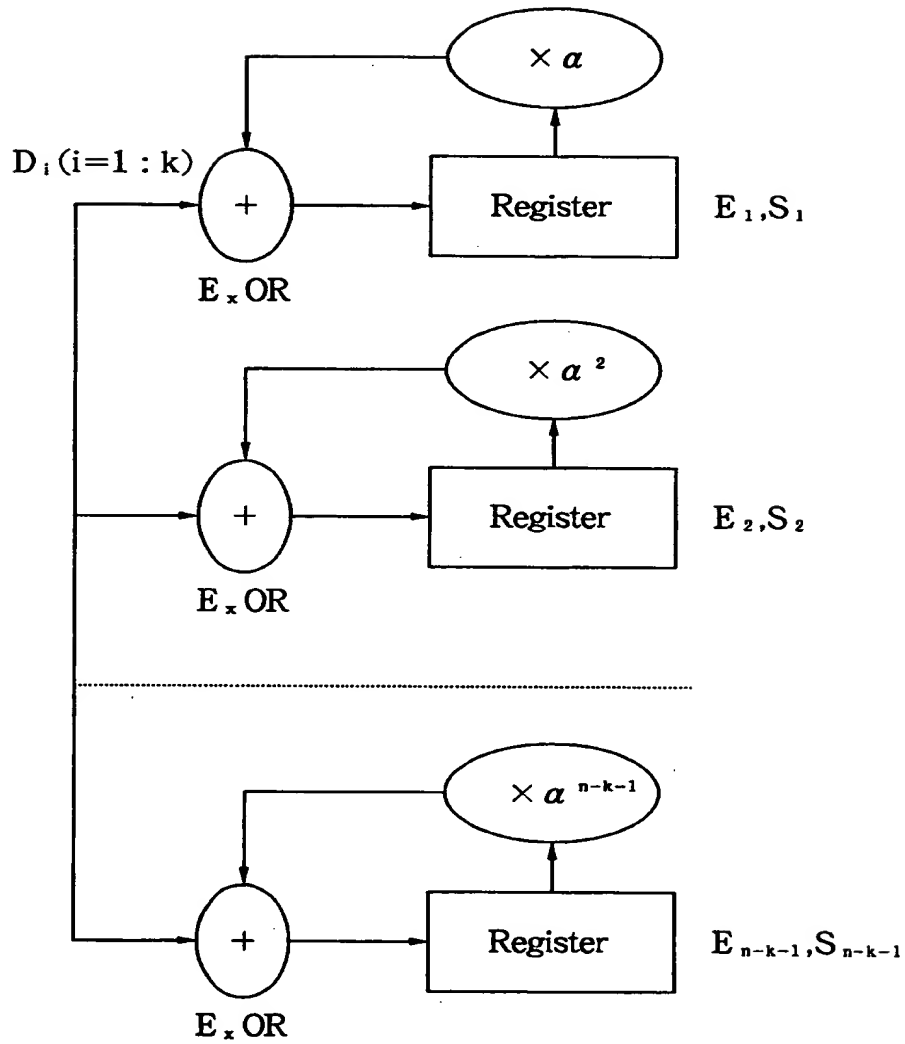
【図 2】

$\alpha$  乗算回路の一例を示すブロック図

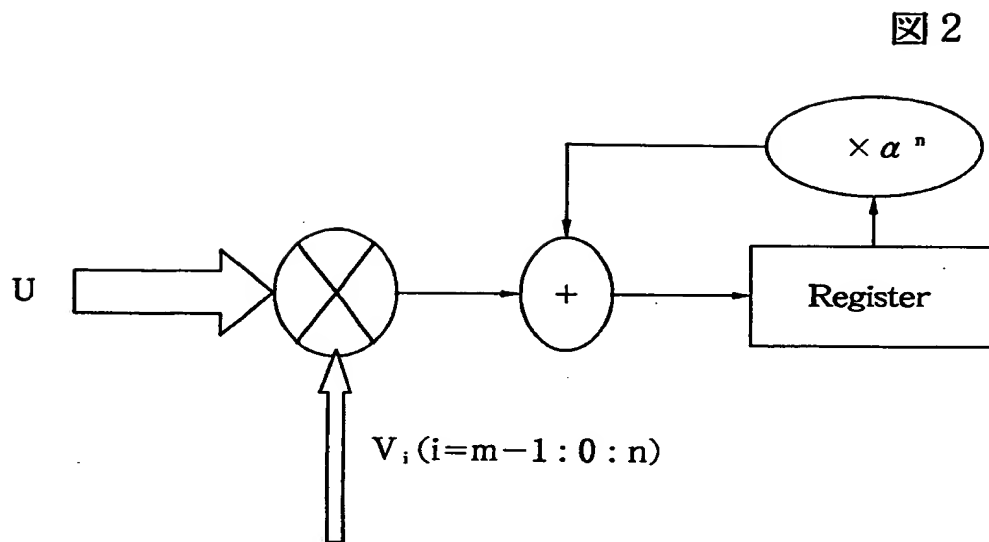
【書類名】 図面

【図 1】

図 1



【図 2】



【書類名】 要約書

【要約】

【課題】 記録媒体にデータをエンコードしまたデコードする場合のシンボルの演算は、ガロア体の原理多項式の元である  $\alpha$  をべき乗算しなくてはならないため、16ビットや32ビットのアクセス幅の場合、参照すべきテーブルの容量が大きくなりすぎる。

【解決手段】 前記  $\alpha$  において、 $\alpha^i$  の演算を行うため、 $i$  ビットのシフト演算部と  $2^i$  の数の参照テーブルを設けた。これにより、小容量のメモリで参照テーブルを実現でき、高速演算が可能となる。

【選択図】 なし

出 願 人 履 歴 情 報

識別番号 [000010098]

1. 変更年月日 1990年 8月27日

[変更理由] 新規登録

住 所 東京都大田区雪谷大塚町1番7号

氏 名 アルプス電気株式会社